

FG

12

AD 1019209

RDA-TR-2701-001

USER'S MANUAL FOR THE COMPUTER PROGRAM ZMODE ✓

JULY 1973

By:  
MICHAEL MILDER

Sponsored By:  
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY  
ARPA Order No. 2239  
Contract No. N00014-73-C-0105  
Program Code No. 3E20  
Name of Contractor: R & D ASSOCIATES  
Principal Investigator and Phone Number: DR. FRANK FERNANDEZ  
451-5838  
Scientific Officer: COMMANDER JOSEPH BALLOU  
Effective Date of Contract: AUGUST 28, 1972  
Contract Expiration Date: OCTOBER 26, 1973  
Amount of Contract: \$249,000.00  
Short Title of Work: OCEAN WAVES AND WAKES

DISTRIBUTION STATEMENT A

Approved for public release;  
Distribution Unlimited

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the United States Government.



R & D ASSOCIATES  
Post Office Box 3580  
Santa Monica,  
California, 90403

525 WILSHIRE BOULEVARD • SANTA MONICA • TELEPHONE: (213) 451-5838

Handwritten signature or initials.

DDC  
RECEIVED  
JAN 12 1976  
RECEIVED  
A

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM	
1. REPORT NUMBER 14 RDA-TR-2701-001	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER 9	4. TYPE OF REPORT & PERIOD COVERED Final Rept.
5. TITLE (and Subtitle) 6 USER'S MANUAL FOR THE COMPUTER PROGRAM ZMODE OCEAN WAVES AND WAKES,		6. PERFORMING ORG. REPORT NUMBER	
7. AUTHOR(S) 10 MICHAEL MILDER		8. CONTRACT OR GRANT NUMBER (if any) 15 N00014-73-C-0105 ARPA Order-2239	
9. PERFORMING ORGANIZATION NAME AND ADDRESS R & D Associates 525 Wilshire Boulevard, P. O. Box 3580 Santa Monica, California 90403		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Code #3E20	
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE 11 July 73	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		13. NUMBER OF PAGES 55	
		15. SECURITY CLASS. (of this report) Unclassified	
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE	
16. DISTRIBUTION STATEMENT (of this Report) The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advance Research Projects Agency or the United State Government.			
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)			
18. SUPPLEMENTARY NOTES			
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)			
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The computer program ZMODE computes the eigenfunctions and dispersion relations for internal wave oscillations in a density-stratified ocean thermocline. The computation is rapid and reliable, owing to a procedure which obtains very accurate trial eigenvalues by exploiting certain analytic properties of the differential eigenfunction equation. This document describes in detail the analytic procedures and program structure, and provides complete operating instructions.			

SUMMARY

The computer program ZMODE computes the eigenfunctions and dispersion relations for internal wave oscillations in a density-stratified ocean thermocline. The computation is rapid and reliable, owing to a procedure which obtains very accurate trial eigenvalues by exploiting certain analytic properties of the differential eigenfunction equation. This document describes in detail the analytic procedures and program structure, and provides complete operating instructions.

## TABLE OF CONTENTS

	<u>Page</u>
1. Introduction . . . . .	1
2. Sample Calculation . . . . .	3
Part I. Operator's Information . . . . .	13
3. Operating Instructions . . . . .	14
4. Operating Tips . . . . .	19
5. Utility Subroutine . . . . .	21
Part II. User's Information . . . . .	23
6. Methods of Calculation . . . . .	24
7. Program Structure . . . . .	29
8. Description of Routines . . . . .	31
a. ZMODE . . . . .	31
b. TCLINE . . . . .	31
c. ZMODES . . . . .	31
d. EIGENVL . . . . .	32
e. EIGENFN . . . . .	34
f. FINT . . . . .	36
g. SPLINES . . . . .	36
h. SPLINE . . . . .	37
i. GRAPH . . . . .	38
9. Listing . . . . .	39

## LIST OF FIGURES

<u>Figure</u>		<u>Page</u>
<u>Sample Calculation</u>		
1	Numerical Thermocline Profile . . . . .	5
2	Graphical Temperature Profile . . . . .	6
3	Graphical Stability Profile . . . . .	7
4	Numerical Dispersion Relations . . . . .	8
5	Graphical Dispersion Frequencies . . . . .	9
6	Mode 1 Eigenfunction Graphical Profile . . . . .	10
7	Mode 2 Eigenfunction Graphical Profile . . . . .	11
8	Mode 3 Eigenfunction Graphical Profile . . . . .	12
<u>Operator's Information</u>		
9	Execution Diagnostics . . . . .	17
<u>User's Information</u>		
10	Program Structure . . . . .	30

# 1. Introduction

The computer code ZMODE calculates the normal mode functions and dispersion relations that characterize the vertical oscillations of a density-stratified ocean thermocline. For an arbitrary input profile of stability frequency,  $N(z)$ , it solves the eigenvalue equation

$$\frac{d^2 f}{dz^2} + k^2 \left( \frac{N^2(z)}{\omega_m^2} - 1 \right) f = 0, \quad (1-1)$$

$$0 \leq z \leq z_b, \quad f(0) = f(z_b) = 0,$$

by direct integration to provide the eigenfrequencies

$$\omega = \omega_m(k), \quad m = 1, 2 \dots \quad (1-2)$$

and associated mode functions  $f$  for up to twenty modes and up to fifty-one equally spaced values of wavenumber  $k$ ,

$$0 \leq k \leq k_{\max},$$

per mode.

The dispersion data are computed and tabulated in the form of inverse phase speeds

$$\gamma_m(k_i) = k_i / \omega_m(k_i), \quad k_i = i \Delta k \quad (1-3)$$

and their derivatives

$$d\gamma_m/dk. \quad (1-4)$$



Because the derivatives are obtained analytically, via properties of equation (1), rather than numerically, the two tables can be used to construct highly accurate interpolated dispersion data. For example, relative interpolation accuracies better than  $10^{-6}$  have been obtained along dispersion curves represented by thirty points.

A particularly efficient and reliable procedure is used for the eigenvalue search, in which the derivatives  $dy_m/dk$  are used to extrapolate successive trial values along each eigenlocus. Converged eigenvalues are consequently obtained in just over two iterations on the average, and the computation time per eigenvalue is (CDC 7600)

$$(\text{eqn (1) integration steps/100}) \times 0.5 \times 10^{-2} \text{ sec.}$$

ZMODE consists of two modules: TCLINE, which converts input temperature-versus-depth data to the stability profile  $N^2(z)$ , and ZMODES, which performs the eigenfunction calculations. The dummy module UTIL is included for the convenience of the user, who can replace it with sub-routines performing various calculations on the quantities provided.

Each module provides its own input and output functions, and operates essentially as an independent program. Input, output and operating parameter specifications are communicated directly to the module named on the control card. While this level of organization may seem a little elaborate, it has been imposed with the idea that ZMODE will ultimately be the nucleus of much larger internal wave codes. It therefore seemed desirable to establish a modular organization and uniform control procedure so that new program modules can be added and used as simply as possible. For this reason also the various kinds of numerical and graphical output are optional, and must be specifically requested by appropriate control cards.

## 2. Sample Calculation

The types of numerical and graphical output available from ZMODE are illustrated in the pages following by reproductions from a test calculation. The shallow-water geometry and thermal scale of this particular case are characteristic of the NUC tower site near San Diego, while the particular double thermocline profile used is a fictitious one designed to exhibit intermode resonances, as will be seen.

Figure 1 shows the numerical output of the interpolated temperature and stability profiles, and Figures 2 and 3 show the graphical counterparts.

The dispersion calculations for this case encompassed modes 1-5, each on a wavenumber range of 0 to 280 cycles/km in 41 increments. Execution time on the CDC 7600 was 1.25 seconds for the dispersion calculations themselves and 1.7 seconds overall. The detailed dispersion printout includes frequency, phase speed and group speed for each mode, as shown in Figure 4.

Graphical display of the dispersion curves is available as in Figure 5. Note the "Eckert Resonance" between modes 2 and 3 at 145 cycles/km, where nearly independent oscillations on the two stable layers become strongly coupled because of an accidental coincidence in frequency. The relative frequency difference at closest approach is 0.013, which is not resolvable on the printer plot.

Printer plots of the eigenfunction profiles, as shown in Figures 6-8, are useful in interpreting thermocline behavior. The six curves shown for each mode correspond to six equally-spaced wavenumbers from zero on the left to maximum on the right.



Note how at high wavenumber the oscillations are confined principally to one of the two stable layers. Note also the sudden exchange occurring in principal layers between modes 2 and 3 at the resonance.

\*\*\* THERMOCLINE PROFILE \*\*\*

DEPTH, M	TEMP, C	N(Z), CY/HR
-0.00	20.00	0.00
-0.40	20.01	0.00
-0.80	20.01	3.80
-1.20	19.99	6.99
-1.60	19.96	8.71
-2.00	19.91	9.60
-2.40	19.86	9.91
-2.80	19.81	10.15
-3.20	19.75	11.37
-3.60	19.67	13.34
-4.00	19.56	15.59
-4.40	19.41	17.24
-4.80	19.24	18.32
-5.20	19.05	19.28
-5.60	18.82	21.90
-6.00	18.50	25.81
-6.40	18.07	27.58
-6.80	17.65	25.08
-7.20	17.31	22.07
-7.60	17.03	20.23
-8.00	16.79	18.48
-8.40	16.60	16.23
-8.80	16.45	13.23
-9.20	16.36	9.74
-9.60	16.32	6.32
-10.00	16.30	3.08
-10.40	16.30	1.72
-10.80	16.29	3.65
-11.20	16.28	6.14
-11.60	16.24	8.71
-12.00	16.17	11.28
-12.40	16.07	13.59
-12.80	15.92	15.20
-13.20	15.75	16.31
-13.60	15.55	16.90
-14.00	15.34	16.76
-14.40	15.15	15.93
-14.80	14.97	14.53
-15.20	14.83	12.99
-15.60	14.71	11.27
-16.00	14.63	9.28
-16.40	14.58	7.51
-16.80	14.54	6.46
-17.20	14.51	6.46
-17.60	14.47	7.17
-18.00	14.43	7.68
-18.40	14.38	7.99
-18.80	14.33	8.13

Figure 1

Numerical Thermocline Profile

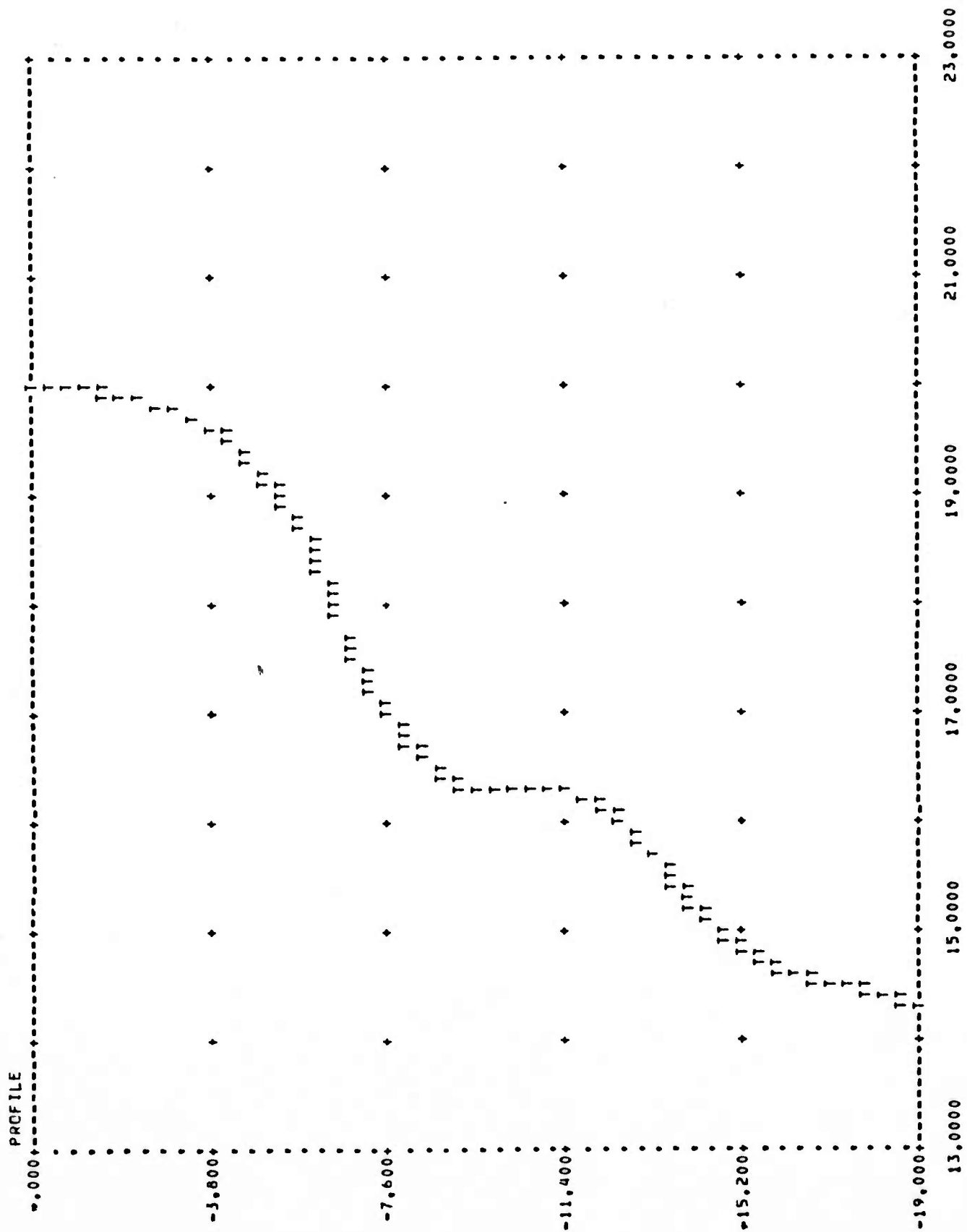


Figure 2. Graphical Temperature Profile

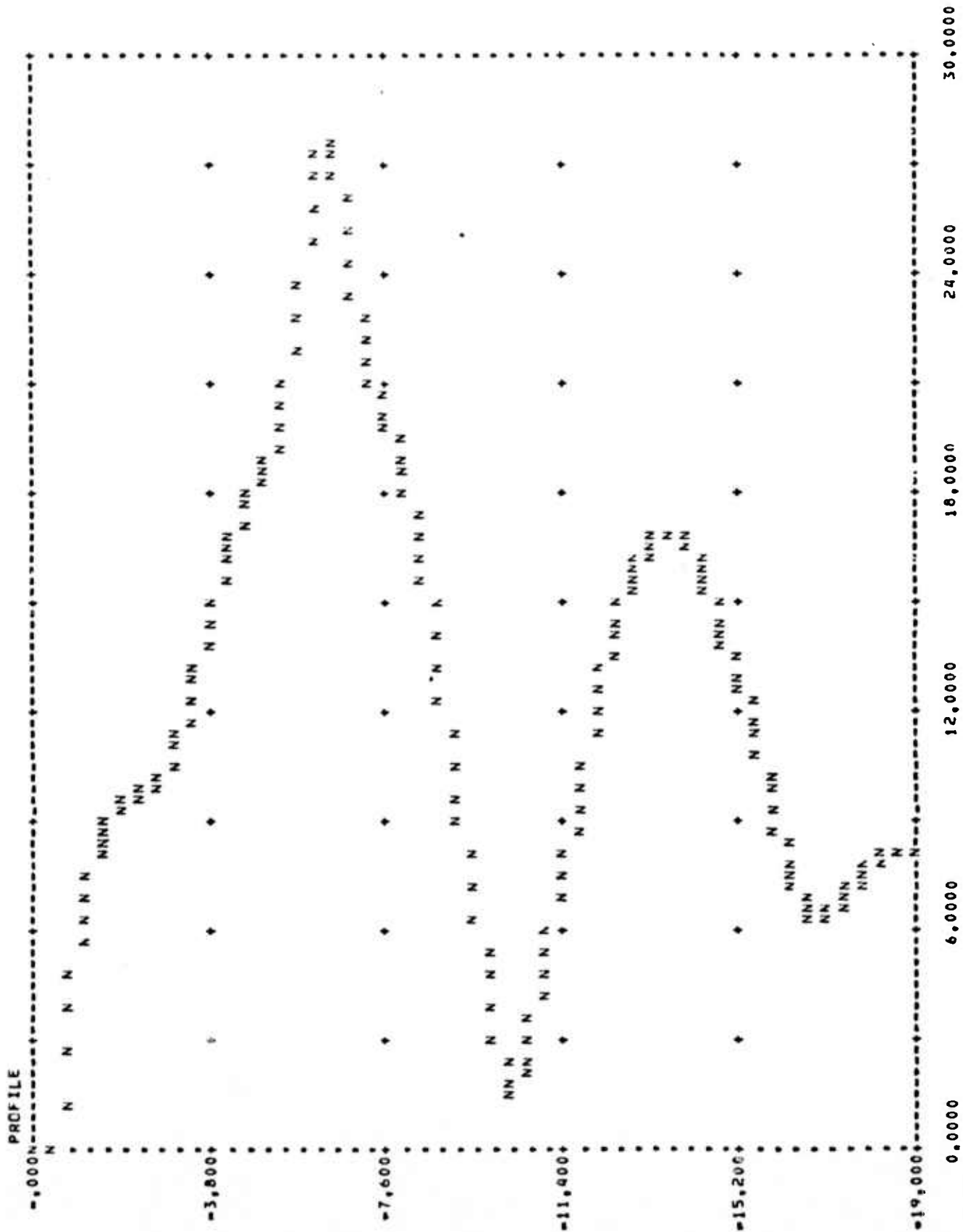


Figure 3. Graphical Stability Profile

\*\*\* MODE 1 DISPERSION RELATIONS \*\*\*

K, CY/KM	W, 1/SEC	W, CY/HR	C, M/SEC	CG, M/SEC
0.00000	0.00000	0.00000	.16912	.16912
7.00000	.00721	4.12816	.16382	.15375
14.00000	.01325	7.59419	.15068	.12014
21.00000	.01780	10.19583	.13487	.08753
28.00000	.02109	12.08173	.11986	.06368
35.00000	.02352	13.47359	.10593	.04792
42.00000	.02538	14.54258	.09618	.03763
49.00000	.02687	15.39788	.08729	.03068
56.00000	.02811	16.10555	.07989	.02575
63.00000	.02916	16.70603	.07366	.02208
70.00000	.03006	17.22531	.06835	.01925
77.00000	.03086	17.68097	.06378	.01700
84.00000	.03157	18.08556	.05981	.01517
91.00000	.03220	18.44831	.05631	.01366
98.00000	.03277	18.77626	.05322	.01240
105.00000	.03329	19.07484	.05046	.01133
112.00000	.03377	19.34837	.04799	.01040
119.00000	.03421	19.60027	.04575	.00961
126.00000	.03462	19.83335	.04372	.00891
133.00000	.03499	20.04991	.04188	.00829
140.00000	.03535	20.25187	.04018	.00775
147.00000	.03568	20.44082	.03863	.00726
154.00000	.03599	20.61815	.03719	.00682
161.00000	.03628	20.78502	.03586	.00643
168.00000	.03655	20.94245	.03463	.00607
175.00000	.03681	21.09130	.03348	.00575
182.00000	.03706	21.23236	.03241	.00545
189.00000	.03729	21.36629	.03140	.00518
196.00000	.03751	21.49369	.03046	.00493
203.00000	.03773	21.61508	.02958	.00470
210.00000	.03793	21.73094	.02874	.00449
217.00000	.03812	21.84168	.02796	.00430
224.00000	.03831	21.94769	.02722	.00412
231.00000	.03848	22.04929	.02651	.00395
238.00000	.03865	22.14679	.02585	.00379
245.00000	.03882	22.24047	.02522	.00365
252.00000	.03897	22.33058	.02461	.00351
259.00000	.03913	22.41735	.02404	.00338
266.00000	.03927	22.50097	.02350	.00326
273.00000	.03941	22.58165	.02298	.00315
280.00000	.03955	22.65955	.02248	.00304

Figure 4. Numerical Dispersion Relations

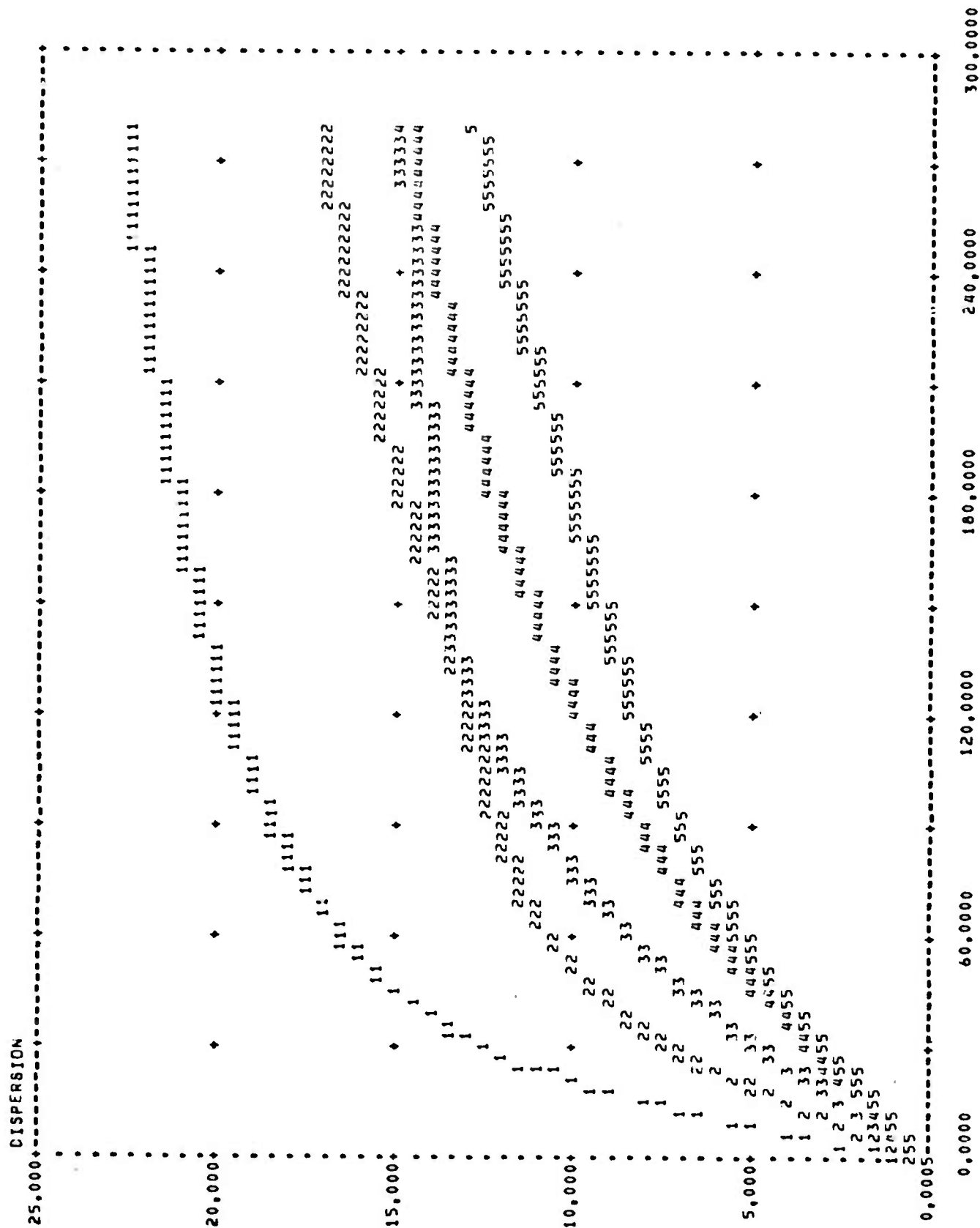


Figure 5. Graphical Dispersion Frequencies



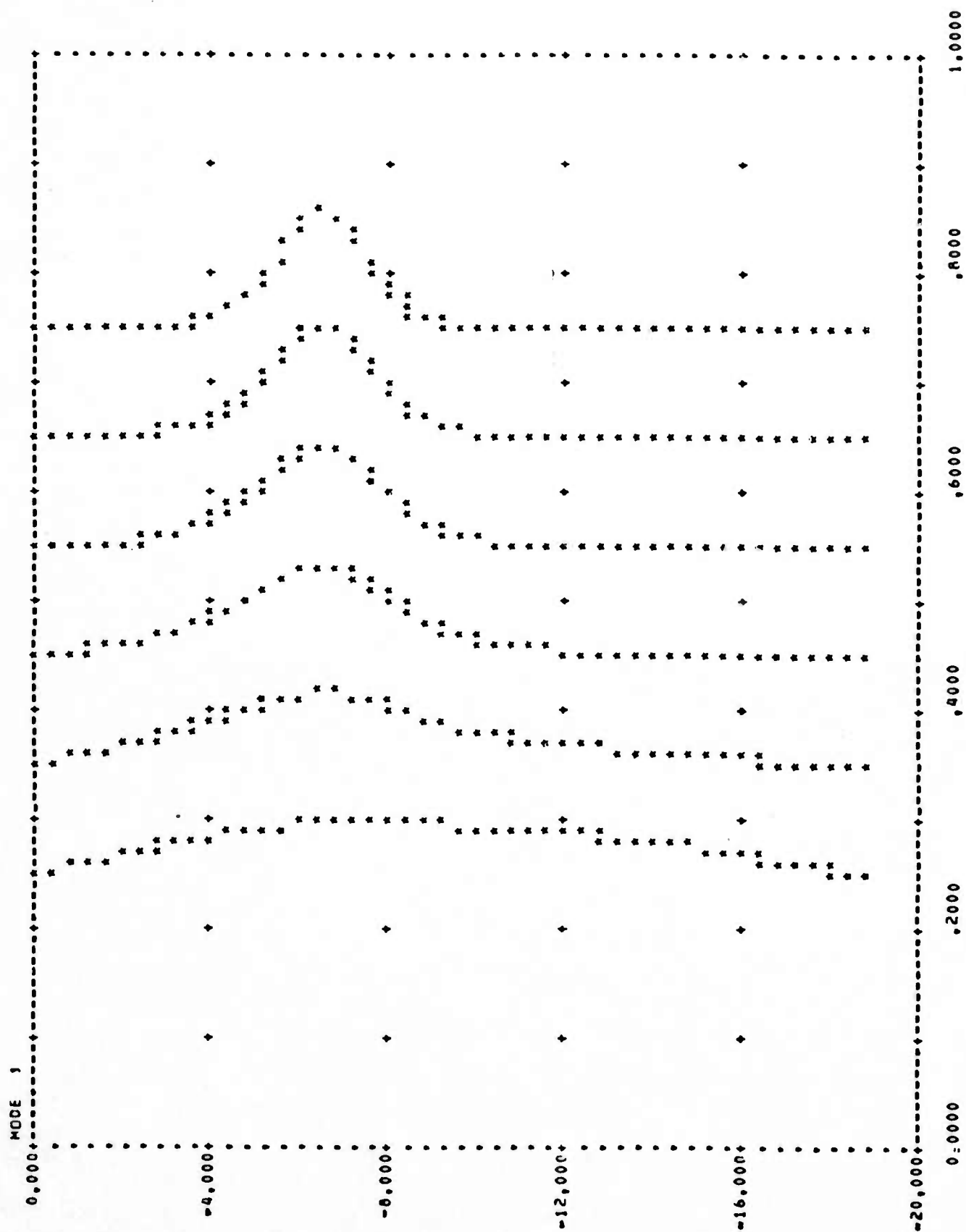


Figure 6  
Mode 1 Eigenfunction Graphical Profile

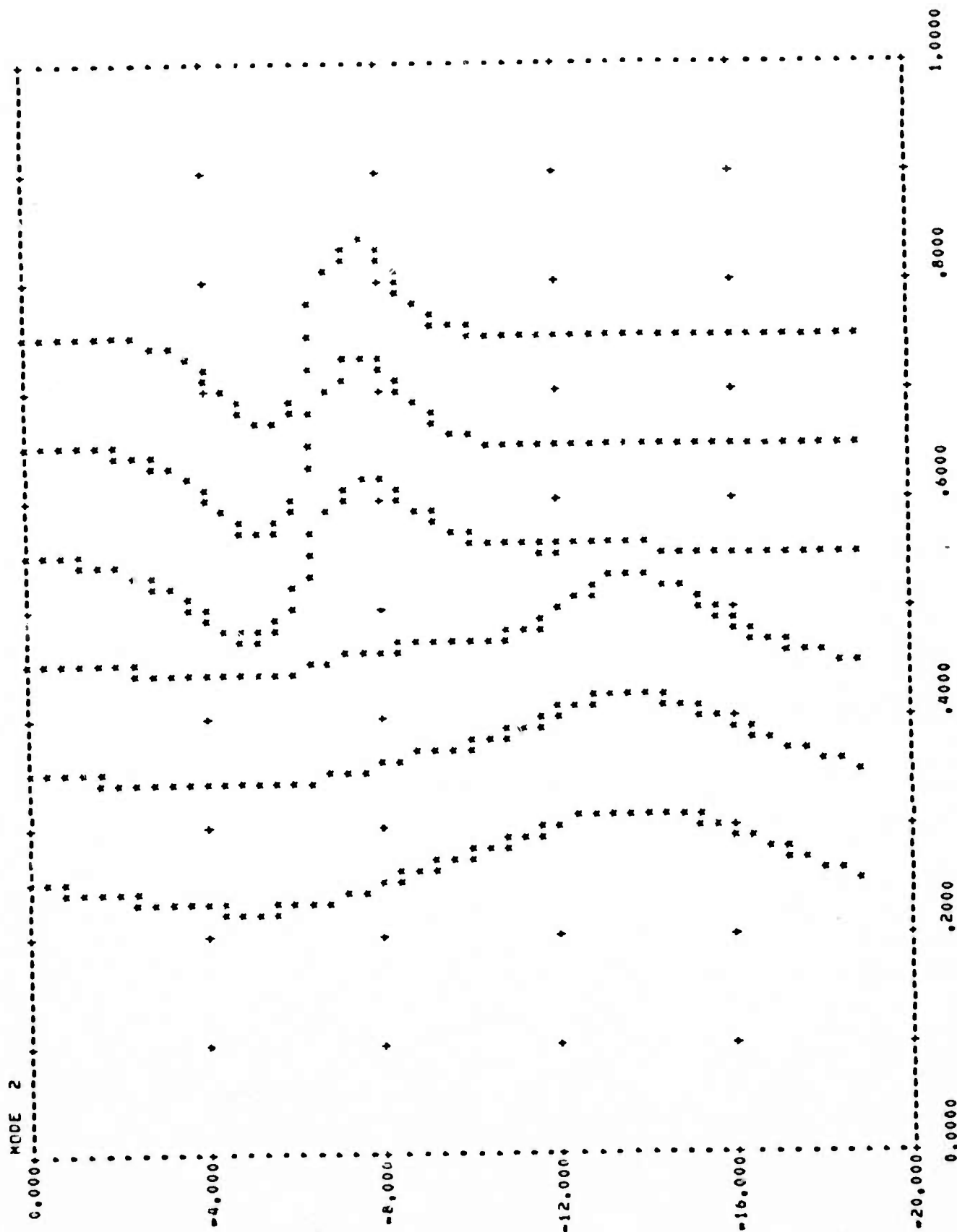


Figure 7  
Mode 2 Eigenfunction Graphical Profile

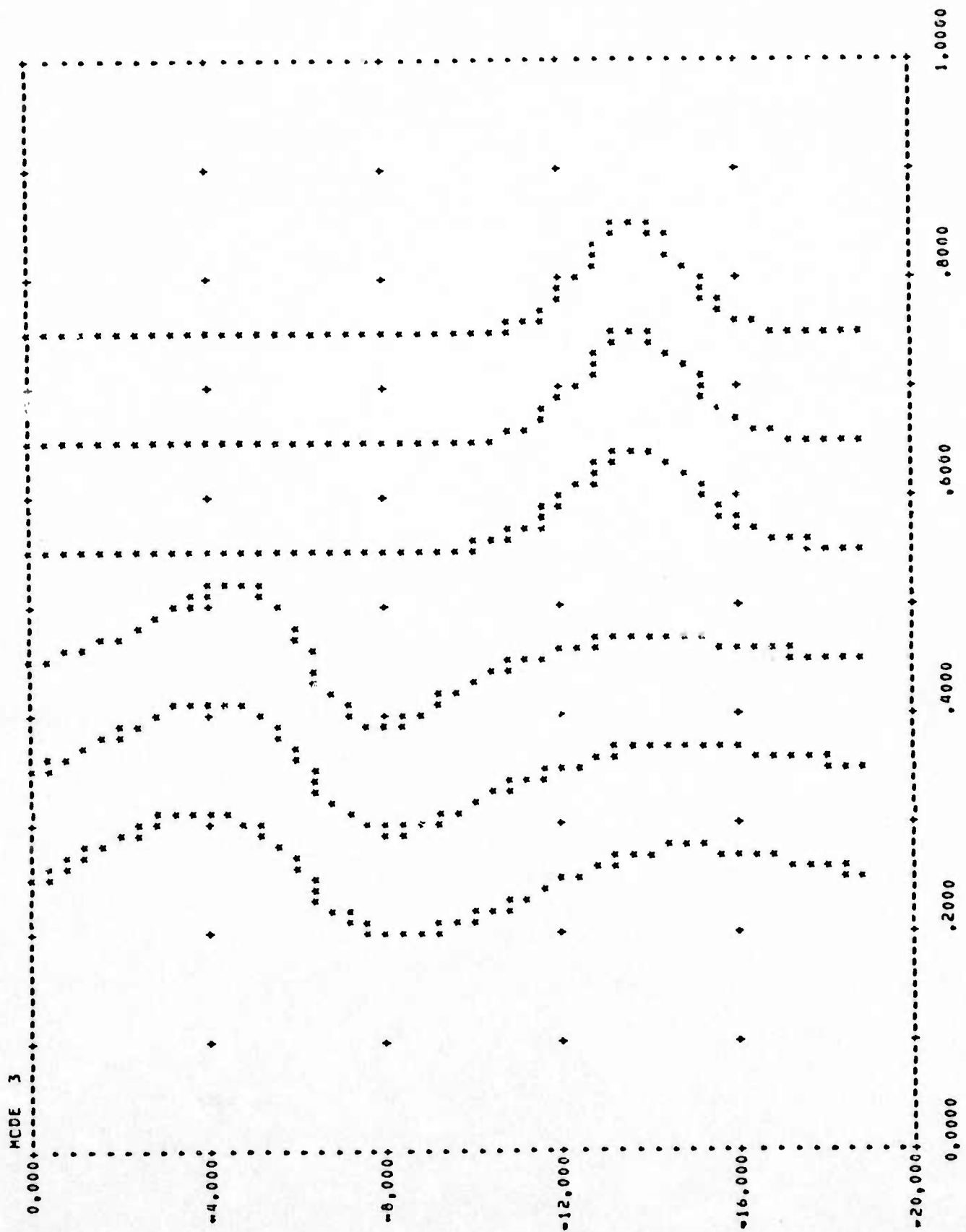


Figure 8  
Mode 3 Eigenfunction Graphical Profile

Part 1.  
OPERATOR'S INFORMATION

### 3. Operating Instructions

All program functions are governed by control cards in a standard format of three ten-character hollerith instruction words (left-justified) followed by up to five ten-character numerical parameters:

$\left\{ \begin{array}{l} \text{INPUT} \\ \text{SET} \\ \text{EXEC} \\ \text{PRINT} \\ \text{GRAPH} \end{array} \right\}$	$\left\{ \begin{array}{l} \text{TCLINE} \\ \text{ZMODES} \\ \text{UTIL} \end{array} \right\}$	$\left\{ \begin{array}{l} \text{(Option} \\ \text{descriptor)} \end{array} \right\}$	$\left\{ \leftarrow \leq 5 \text{ parameters} \rightarrow \right\}$
---------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------	---------------------------------------------------------------------

The first word describes the function to be performed and the second describes the module to which the instruction is addressed. The instructions are self-explanatory; the distinction between INPUT and SET is that INPUT refers to tabular data and SET refers to adjustable program parameters. Cards referring to different modules can be intermixed, the only constraints being the obvious ones that INPUT and SET must precede the first EXEC for a given module, and PRINT and GRAPH must follow. EXEC TCLINE must precede EXEC ZMODES. Any number of control cards can be used. For example, repeated calculations with varying thermocline data, wavenumber increments, or wavenumber range can be requested, and duplicate output for a given calculation can be obtained with multiple output cards.

Cards with arbitrary alphanumeric content placed before the first control card will be duplicated on the output title page and otherwise ignored; these can be used as desired for descriptive titles. Execution is terminated by a card with a left-justified STOP. The various control parameters are described below.

INPUT TCLINE:        (NDATA)  
Format                ( I10 )

This instruction is followed by cards containing the temperature-versus-depth data in order of increasing depth,

$Z_i, T_i, i = 1, 2, \dots, \text{NDATA}$         units: Meters, Celcius

in the format (2F10.\_).  $Z_1$  must be zero.  $Z(\text{NDATA})$  must be greater than or equal to the parameter ZA, described in the following. NDATA must be in the range (3,100).

SET TCLINE:        (ZA, ZB, N )  
Format                (2F10.\_, I10)

ZA is the depth (M) of the thermocline bottom, below which it is assumed that density stratification vanishes; ZA must be less than or equal to the depth of the last input data point. ZB (M) is the depth of the ocean bottom, greater than or equal to ZA. The parameter N defines the number of integration steps between zero and ZA used in the mode calculations and is specified here so that the interpolated stability-frequency table created by TCLINE will have the right-sized increments. A useful rule of thumb is that N should exceed ten times the highest mode order to be calculated; maximum permissible value is 200.

EXEC TCLINE:        (No parameters)  
PRINT TCLINE:        (ISKIP)  
Format                ( I10 )

Up to 399 values (i.e.,  $2N-1$ ) of interpolated temperature and stability frequency can be printed. For shortened printout, only every ISKIP<sub>th</sub> value is printed. When ISKIP is blank the default value 4 is used. See Figure 1, p.5.



GRAPH TCLINE: (T1, T2, WSCALE)  
Format ( 3 F10.\_\_\_\_ )

This produces a printer plot of T in the interval (T2, T1) versus Z in the interval (0,ZA), and a plot of N(Z) in the interval (0, WSCALE) versus Z. Units are meters, C, and cy/hr. See the example in Figures 2 and 3, pp. 6&7. If the proper value of WSCALE is not known, a rough value can be estimated and several values tried at once with repeated GRAPH TCLINE cards.

INPUT ZMODES: (Not Used)  
SET ZMODES: (KMAX, NK, ERR)  
Format (F10.\_\_\_\_, I10, E10.0)

This prepares ZMODES to find eigenfunctions and eigenvalues at NK ( $\leq 51$ ) equally-spaced values of wavenumber k in the interval  $0 \leq k \leq KMAX$ . The units of KMAX are cycles/km, converted internally to radians/m. ERR specifies the minimum relative precision to which the iterated eigenvalue search must converge. The default value is  $10^{-8}$ , but up to  $10^{-13}$  is usable on a 60 bit machine.

EXEC ZMODES: (M1, M2, Z1, Z2)  
Format (2I10, 2F10.\_\_\_\_)

The above instructs ZMODES to find eigenfunctions and eigenvalues for modes M1 through  $M2 \leq 20$ . As calculations are completed for each mode a short line of diagnostic output is produced listing the required computation time and average number of search iterations per eigenvalue. See Figure 9. Two depths, Z1 and Z2, can be specified in the interval (0,ZA), at which eigenfunction values and Z-derivatives will be tabulated along with dispersion data during execution of ZMODES.

\*\*\* EIGENVL COMPUTATION TIMES \*\*\*

\* RESOLUTION 41 VALUES/MODE \*

MODE	AVG ITERATIONS	TIME
1	2,71	,2350
2	2,83	,2480
3	3,20	,2760
4	2,66	,2340
5	3,12	,2690

Figure 9  
Execution Diagnostics

PRINT ZMODES: (M1, M2)  
Format ( 2I10 )

Provides printed tabular output, versus wavenumber, of frequency, phase speed, and group speed, for modes M1 through M2. See Figure 4, p.8.

GRAPH ZMODES } FUNCTIONS { M1, M2, WSCALE, KSCALE  
                  } M1, M2, MULT, FSCALE, ZSCALE )  
Format ( { 2 } I10, 2F10. )  
          { 3 }

With the blank option this instruction plots the frequency versus wavenumber dispersion curves for modes M1 through M2 as shown in Figure 5, P. 9. Plot ranges are (0, KSCALE) cy/km in wavenumber and (0, WSCALE) cy/hr in frequency.

The FUNCTIONS option plots the eigenfunctions themselves side by side for MULT ( $\geq 2$ ) equally spaced wavenumbers between 0 and KMAX, one mode per figure for modes M1 through M2. Figures 6-8, PP.10-12, show examples for modes 1-3 with MULT=6.

The functions are automatically normalized for plotting and the dimensionless number FSCALE is available for further scaling. A fair rule of thumb for FSCALE is

$$FSCALE \sim MULT \times M2$$

but this can vary with thermocline character.

#### 4. Operating Tips

Stable eigenvalue searches along the eigenloci require that the extrapolated eigenvalue predictions be accurate relative to the intermode eigenvalue spacing. At large wavenumbers this stability can be jeopardized by two effects: long exponential scales in the eigenfunction integration introduce numerical noise into the calculation of eigenlocus slope and degrade the extrapolation accuracy; so-called "Eckart resonances" between isolated stable layers become severe and intermode eigenvalue spacing can become exponentially small. In this context wavenumbers are "large" or "small" in the comparison

$$KMAX > \frac{1000}{ZA} \quad .$$

Convergence precision (ERR) of  $10^{-6}$  -  $10^{-8}$  will probably be adequate for small wavenumbers, and  $10^{-10}$  for the larger wavenumbers of practical interest. In limited tests and ERR setting of  $10^{-13}$  removed all evidence of numerical noise out to  $KMAX \cdot ZA = 10,000$ .

Since the error in the quadratic eigenvalue extrapolation scales as  $NK^{-3}$ , the ability of the extrapolation to get through an Eckart resonance can be improved sharply by an increase in NK.

An accidental convergence to the wrong mode will result in program termination with the error message "EIGENVL failed to converge at mode \_\_\_\_\_, wavenumber \_\_\_\_\_."

The average number of iterations per eigenvalue necessary to converge to ERR, printed for each mode during ZMODES execution, is an indication of the quality of extrapolations being obtained; less accurate trial eigenvalues require more iterations. The number should be just above two for small KMAX, large NK, and  $ERR \sim 10^{-8}$ . Values around three are to be expected for  $ERR < 10^{-10}$ . Since total execution time is proportional

to NK times the average number of iterations, the smallest NK and largest ERR consistent with reliable computation should be aimed for when computing expense is a factor.

## 5. Utility Subroutine

Various quantities calculated by ZMODE are available to the user in the dummy subroutine UTIL:

<u>Quantity</u>	<u>Symbol</u>	<u>FORTRAN</u>
Wavenumber	$k$	TK (IK)
Inverse phase speed	$\gamma_m(k)$	TGAMMA (IK, MODE)
Associated derivative	$d\gamma_m/dk$	TDGAMMA (IK, MODE)
Mode amplitude	$\phi_m(z_j, k)$	FI (IK, MODE, JZ)
Associated derivative	$\partial\phi_m/\partial z$	FIZ (IK, MODE, JZ)
Mode sample depths	$z_j, j = 1, 2$	Z1, Z2
Table lengths	(IK = 1, NK)	NK

From the first three tables one can obtain a cubic interpolation of  $\gamma_m$  for any  $k$ . Related dispersion quantities can then be computed via

$$c_m = \gamma_m^{-1} \quad (\text{phase speed})$$

$$\omega_m = k\gamma_m^{-1} \quad (\text{frequency})$$

$$c_{gm} = d\omega_m/dk$$

$$= c_m (1 - \omega_m d\gamma_m/dk) \quad (\text{group speed});$$

the units are (meters, seconds, radians).



The mode amplitude and derivative data are retained at only two depths, Z1 and Z2, as specified on the EXEC ZMODES card. These are intended for use in constructing two-point Green's functions needed in studies of internal wave generation. Another possible application is the prediction of surface current components  $u_m(0, k)$  associated with measured isotherm displacements  $\zeta_m(z_1)$  at a given depth  $z_1$ :

$$u_m = \left[ \gamma_m^{-1} \phi_m^{-1}(z_1) \frac{d\phi_m(0)}{dz} \right] \zeta_m(z_1). \quad (5-1)$$

Part II.

USER'S INFORMATION

## 6. Methods of Calculation

The thermocline is numerically represented by a table of stability frequency  $N^2(z)$  for  $2n-1$  equally-spaced depths spanning the interval  $(-z_a, 0)$ ; outside this interval, that is, in the remaining interval  $(-z_b, -z_a)$  to the ocean bottom,  $N^2(z)$  is assumed to vanish. The numbers  $z_a$ ,  $z_b$  and  $n$  are input parameters. The table is obtained from an input table of temperature data,  $T(z_i)$ , by a third-order spline interpolation and differentiation, via the formula

$$N^2(z) = - \alpha(T) g \frac{dT}{dz} \quad (6-1)$$

in which  $\alpha(T)$  is a linear representation of the coefficient of thermal expansion of seawater at 35% salinity. The use of the third-order spline yields a smooth stability profile with a continuous derivative.

The eigenvalue equation

$$\frac{d^2 f}{dz^2} + (\gamma^2 N^2 - k^2) f = 0, \quad (6-2)$$

here written in terms of wavenumber  $k$  and inverse phase speed  $\gamma$  as parameters, has solutions that satisfy the boundary conditions

$$f(-z_b) = f(0) = 0 \quad (6-3)$$

on an infinite number of real loci  $\gamma_m(k)$ . At each  $k$ , the eigensolutions obey the orthogonality rule

$$\int_{-z_b}^0 f_m N^2 f_n dz = \mu_m \delta_{mn}. \quad (6-4)$$

The numerical search for the eigenfunctions  $f_m$ , eigenvalues  $\gamma_m$ , and normalizing factors  $\mu_m$  proceeds as follows. For a given value of  $k$  and a trial value of  $\gamma$  assumed close to the  $m^{\text{th}}$  eigenlocus, the trial function

$$f = f(z, \gamma, k), \quad f(-z_a) = 0$$

is numerically integrated in  $n$  steps from  $-z_a$  to 0, the appropriate starting values of  $f_m(-z_a)$  and  $f'_m(-z_a)$  having been found from the known analytic solution

$$f = \text{const.} \times \sinh k(z + z_b) \quad (6-5)$$

in the region  $(-z_b, -z_a)$  where  $N^2 = 0$ . The value of  $f$  at  $z = 0$ ,

$$w(\gamma, k) \equiv f(0, \gamma, k) \quad (6-6)$$

is, in general, not zero and vanishes only when  $\gamma = \gamma_m(k)$ . The quantity  $w$  is a continuous, differentiable function of  $\gamma$  and  $k$ , and its properties are key to the eigenvalue search procedure. If we define

$$g(z, \gamma, k) \equiv \frac{\partial f}{\partial \gamma^2}(z, \gamma, k) \quad (6-7)$$

and formally differentiate the eigenvalue equation with respect to  $\gamma^2$ , we see that  $g$  satisfies the inhomogeneous equation

$$\frac{d^2 g}{dz^2} + (\gamma^2 N^2 - k^2)g = -N^2 f, \quad (6-8)$$

$$g'(-z_a) = g'(-z_a) = 0.$$

This companion equation is simultaneously integrated with (6-2), yielding

$$\frac{1}{2\gamma} \frac{\partial w}{\partial \gamma} = g(0, \gamma, k), \quad (6-9)$$

which can be used immediately to generate an improved estimate of  $\gamma$ ,

$$\gamma \rightarrow \gamma - \left( \frac{\partial w}{\partial \gamma} \right)^{-1} w. \quad (6-10)$$

A few such iterations drive  $w$  quickly to zero, within the limit of computational precision. The numerical integration of the similar equations (6-2, 6-8) uses constant step size and identical algorithms. Since the resulting finite difference equation for  $g$  is, in fact, the formal derivative of the finite difference equation for  $f$ , the convergence is very smooth, and the correction in (6-10) will approach within a few bits of zero.

The value of  $\partial w / \partial \gamma$  has, upon convergence to the eigenvalue, the following significance. Multiplying (6-2) by  $g_m$ , (6-8) by  $f_m$ , and subtracting, we have

$$\frac{d}{dz} \left( g_m \frac{df_m}{dz} - f_m \frac{dg_m}{dz} \right) = -N^2 f_m^2; \quad (6-11)$$

integrating this expression and remembering that  $f_m$  vanishes at  $-z_a$  and 0, while  $g$  vanishes at  $-z_b$ , we find

$$\frac{1}{2\gamma_m} \left( \frac{\partial w}{\partial \gamma} \right) f'_m(0) = - \int_{-z_b}^0 N^2 f_m^2 dz = -\mu_m. \quad (6-11)$$

Thus, the calculated quantity  $\partial w / \partial \gamma$  furnishes the normalization constant  $\mu_m$ . An analogous argument shows that

$$\frac{1}{2k} \left( \frac{\partial w}{\partial k} \right) f'_m(0) = \nu_m \quad (6-12)$$

where  $\nu_m$  is the simple norm

$$\nu_m = \int_{-z_b}^0 f_m^2 dz \quad (6-13)$$

Upon completion of each iteration, this last quantity is calculated by direct integration of the squared eigenfunction. The slope of the eigenlocus,  $d\gamma_m/dk$ , defined by

$$\frac{\partial w}{\partial \gamma} d\gamma_m + \frac{\partial w}{\partial k} dk = 0,$$

is then obtained as

$$\frac{d\gamma_m}{dk} = - \frac{\partial w / \partial k}{\partial w / \partial \gamma_m} = \frac{k \nu_m}{\gamma_m \mu_m}. \quad (6-14)$$

Formally, the identity above is a differential equation for the entire  $m^{\text{th}}$  eigenlocus. Numerically, it is used to extrapolate accurate trial eigenvalues at successive points along the locus,



$$\gamma_m(k+\Delta k) = \gamma_m(k) + \left[ \frac{3}{2} \frac{\partial \gamma_m}{\partial k}(k) - \frac{1}{2} \frac{\partial \gamma_m}{\partial k}(k-\Delta k) \right] \Delta k + O(\Delta k)^3. \quad (6-15)$$

Starting trial eigenvalues for all the loci at  $k=0$  are provided by the WKB approximation to (6-2).

$$\gamma_m(0) \int_{-z_a}^0 N(z) dz \cong \left(m - \frac{1}{2}\right) \pi; \quad (6-16)$$

the first extrapolation from  $k=0$  is modified from (6-14) to use the second derivative

$$\frac{d^2 \gamma_m(0)}{dk^2} = \frac{v_m}{\gamma_m \mu_m}$$

obtained from (6-13) by L'Hopital's rule.

## 7. Program Structure and Nomenclature

The subroutine calling sequence and data flow are depicted in Figure 10. The output data tables comprising COMMON/FLD/ have already been described in section 5. Other variables and working files are:

Description	Symbol	FORTRAN
Integration step size	$dz$	DZ
Number of steps	$n$	N
Stability profile	$N^2(z)$	QN(I), I=1, N2
	$2n-1$	N2
Trial eigenfunction	$f(z, \gamma, k)$	$F(I) \left\{ \begin{array}{l} I=1, N \end{array} \right.$
	$(\partial f / \partial z) \cdot dz$	$FZ(I) \left\{ \begin{array}{l} I=1, N \end{array} \right.$
Associated function	$g(z, \gamma, k) / dz^2$	$G(I) \left\{ \begin{array}{l} I=1, N \end{array} \right.$
	$(\partial g / \partial z) / dz$	$GZ(I) \left\{ \begin{array}{l} I=1, N \end{array} \right.$
Wavenumber	$k$	K (real)
Increment	$\Delta k$	DK
Trial phase slowness	$\gamma$	GAMMA
Temperature profile	$T(z)$	$TDATA(I) \left\{ \begin{array}{l} I=1, NDATA \end{array} \right.$
		$ZDATA(I) \left\{ \begin{array}{l} I=1, NDATA \end{array} \right.$
Mode number	$m$	MODE

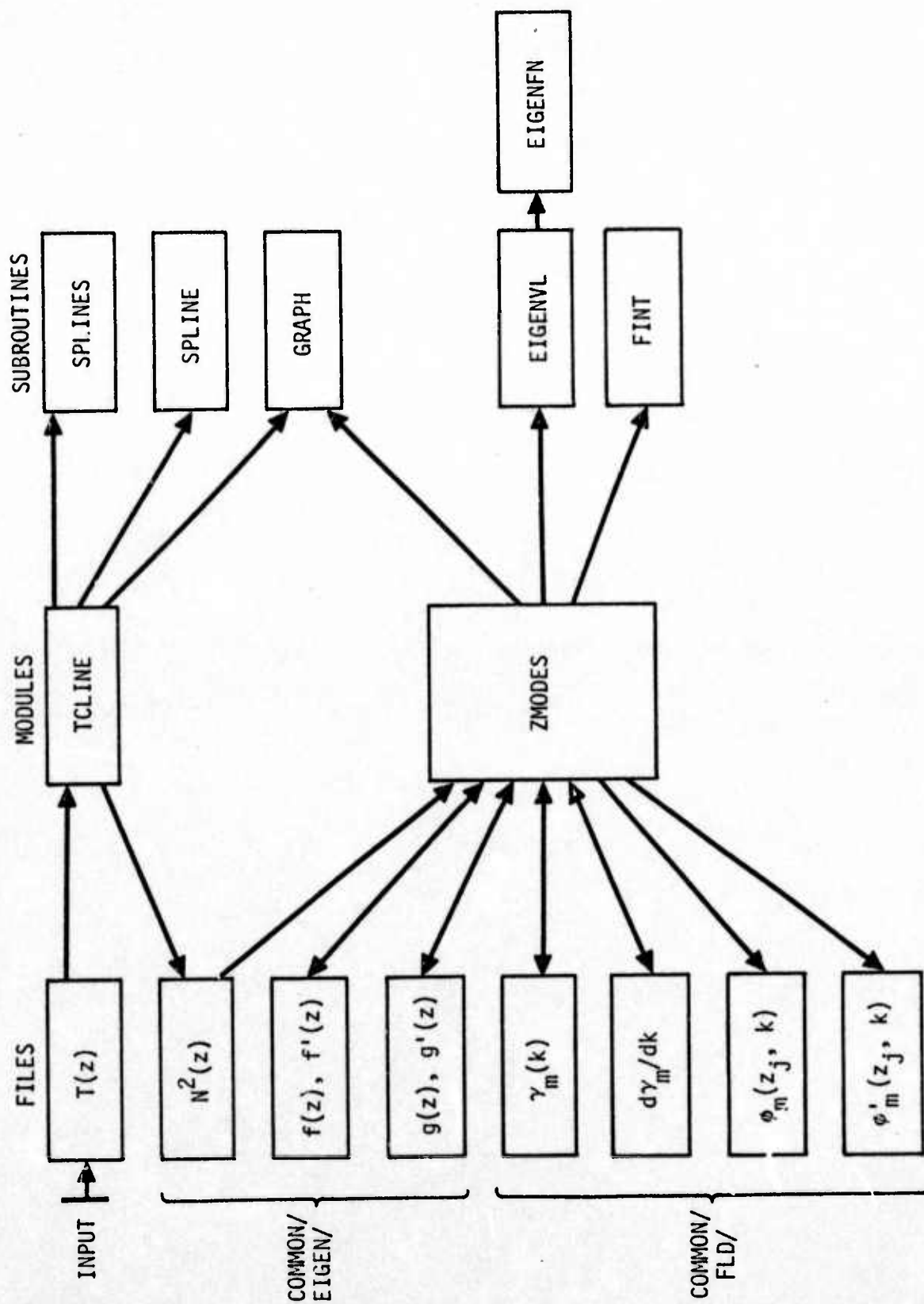


Figure 10. Program Structure

## 8. Description of Routines

### a. ZMODE

ZMODE is the driver, whose only function is to pass control-card images to the module designated: TCLINE, ZMODES, or UTIL.

### b. TCLINE

TCLINE reads the temperature-versus-depth data, calls SPLINES to compute the cubic spline coefficients, then uses SPLINE (ENTRY SPLINE2) to obtain interpolated values of  $dT/dz$  at  $2n-1$  equally-spaced points between  $-z_a$  and 0. The stability profile is computed and tabulated as

$$N^2(z) = -g(c_1 + c_2 T) \frac{dT}{dz} \quad (8-1)$$

in which  $c_1 + c_2 T$  is a linear fit to the coefficient of thermal expansion of water at 35% salinity. TCLINE also computes the quantity.

$$HN = \int_{-z_a}^0 N(z) dz \quad (8-2)$$

for later use by ZMODES in obtaining starting eigenvalues via the WKB approximation (see eqn. 6-16).

### c. ZMODES

ZMODES computes the dispersion tables  $\gamma_m(k)$  and  $d\gamma_m/dk$  for all modes in the interval (M1, M2) designated on the execution card.

For each mode, ZMODES computes a starting trial eigenvalue for  $k = 0$  via the WKB approximation, calls EIGENVL to compute  $\gamma_m$  and  $d\gamma_m/dk$  to the specified precision, and thereafter repeatedly increments  $k$ , calls EIGENVL,

and tabulates  $\gamma_m$ ,  $d\gamma_m/dk$  up to the specified wavenumber limit. Successive trial eigenvalues are computed according to the scheme described in section 6 and passed to EIGENVL with each call. ZMODE will terminate execution with an error message if the diagnostic integers IERR and MERR returned by EIGENVL are not both zero (see below).

For the printer plot of the dispersion curves  $\omega_m(k)$  an auxiliary 200-point table is computed in convenient cy/hr units from cubic interpolations of  $\gamma_m(k)$  provided by FINT. For the printer plots of the unnormalized eigenfunctions a scale factor derived from

$$v_m = \int f_m^2 dz$$

is passed to the plotting subroutine GRAPH.

d. EIGENVL

This subroutine provides the iteration logic for the convergent eigenvalue search at a given mode and wavenumber, and computes the eigenfunction norm parameters used by ZMODES for trial eigenvalue extrapolation. The argument list is

GAMMA	Inverse phase speed $\gamma$ ; trial value on input, converged value on return
K	Wavenumber $k$
NORM	Norm $v$
ONORM	Weighted norm $\mu$
MODE	Desired mode number

MERR	Mode error flag
IERR	Iteration failure flag
IT	Number of iterations required

EIGENVL repeatedly calls EIGENFN to perform integrations of the eigenvalue equation with the current value of  $\gamma$ , each time improving  $\gamma$  by Newton's method

$$\gamma \rightarrow \gamma - \Delta\gamma, \quad \Delta\gamma = \left( \frac{dw}{d\gamma} \right)^{-1} w$$

until  $w$  is sufficiently close to zero that

$$|\Delta\gamma/\gamma| < \text{ERR}.$$

Accidental convergence to an adjacent mode is prevented by a check of the sign of  $dw/d\gamma$ , which alternates as a function of mode number. If the sign is wrong, or if for some reason convergence has not been obtained after ten iterations, IERR is changed from 0 to 1.

The number of zero crossings,  $M$ , occurring in  $f(z)$  (excepting  $z=0$ ) is counted and the number

$$\text{MERR} = M - \text{MODE} + 1$$

formed; if either  $\text{IERR} \neq 0$  or  $\text{MERR} \neq 0$  further computation is bypassed.

After normal convergence EIGENVL makes a linear correction to the eigenfunction using the last computed value of  $\Delta\gamma$ ,

$$f(z) \rightarrow f(z) - 2\gamma\Delta\gamma g(z). \quad (8-3)$$

The reason for this correction is that a small relative error in eigenvalue,  $\Delta\gamma/\gamma$ , can be exponentially amplified by the eigenvalue equation in regions where  $\omega_m^2 > N^2(z)$ , leading to errors in  $f_m$  of order

$$\frac{\delta f}{f} \sim \frac{\Delta\gamma}{\gamma} e^{kz_a},$$

so that, for example, at  $kz_a \sim 30$  the error can grow to order unity even for  $\Delta\gamma/\gamma = 10^{-13}$ . Fortunately, the error is predictable and removable by (8.3), to order

$$\frac{\delta f}{f} \sim \left(\frac{\Delta\gamma}{\gamma}\right)^2 e^{kz_a} \quad (8-4)$$

The norm quantity  $\mu_m$  is then computed via

$$\mu_m = -\frac{1}{2\gamma_m} \frac{\partial W}{\partial \gamma_m} f'_m(0),$$

and the quantity  $v_m$  by integration,

$$v_m = \int_{-z_b}^0 f_m^2 dz,$$

analytically in the region  $(-z_b, -z_a)$ , and numerically according to the end-corrected trapezoidal rule in the region  $(-z_a, 0)$ , with a small predicted error of order

$$z_a (f^2)^{(4)} (dz)^4$$

e. EIGENFN

EIGENFN integrates  $f$  and the associated function  $g$  on the interval

$(-z_a, 0)$ , using a fourth - order algorithm furnished by G. Peebles of RDA:

$$\bar{y}_{i+k_2} = y_i + \frac{1}{2} \Delta z y'_i + \frac{1}{8} \Delta z^2 y''_i$$

$$\bar{y}''_{i+k_2} = y''(\bar{y}_{i+k_2})$$

$$y_{i+k_2} = \bar{y}_{i+k_2} + \frac{1}{24} \Delta z^2 (\bar{y}''_{i+k_2} - y''_i)$$

$$y''_{i+k_2} = y''(y_{i+k_2})$$

$$y_{i+1} = y_i + \Delta z y'_i + \frac{1}{6} \Delta z^2 (y''_i + 2y''_{i+k_2})$$

$$y''_{i+1} = y''(y_{i+1})$$

$$y'_{i+1} = y'_i + \frac{1}{6} \Delta z (y''_i + 4y''_{i+k_2} + y''_{i+1})$$

A constant step size  $\Delta z$  is used to allow convenient non-dimensionalization, with the consequent savings in arithmetic.

The arguments

$$W = -f(0)$$

$$WG = -g(0)$$

are set prior to return.



f. FINT

The function subroutine FINT performs a cubic interpolation of a tabulated function with the aid of corresponding tabulated derivative. The arguments are

Z	Interpolation point
ZT	Array containing values of independent variable Z
FT	Array containing values of tabulated function
FTZ	Array containing values of tabulated derivative
N	Dimension of arrays
I	Pointer index

The subroutine logic finds I such that  $ZT(I) \leq Z < ZT(I+1)$  prior to interpolation, and the value of I can be held for subsequent calls to minimize table searches.

g. SPLINES

SPLINES computes third-order spline coefficients for use by the interpolating subroutine SPLINE. The arguments are

N	Number of data ( $\leq 100$ )
Z	Array containing values of independent variable
C	Array containing values of dependent variable
A	Array containing computed spline coefficients

The spline coefficients are equal to one-sixth the second derivative and are calculated by an efficient method due to L. Solomon of Planning Systems, Inc.  $A(1)$  and  $A(N)$  are assumed to be zero.

h. SPLINE

Spline provides cubic interpolation on the basis of tabulated values of a function and accompanying spline coefficients. The arguments are

Z	Interpolation point
F	Interpolated function value
FZ	Interpolated derivative value
ZT	Arrays containing values of the independent variable, dependent variable, and spline coefficients
FT	
A	
N	Number of points
I	Pointer index

The pointer index is used as in FINT. The calculation of the derivative is ordinarily bypassed to save time; it is implemented via ENTRY SPLINE2.

i. GRAPH

This printer-plot subroutine depends on word size and bit-manipulating subroutines specific to the CDC 6600 and 7600, and will have to be replaced for different machines. The argument list is straightforward -

X, Y	Arrays containing values to be plotted
X0, Y0	Minimum values, lower left
XSCALE, YSCALE	Axis ranges
N	Number of points to be plotted
MODE	See below
SYMBOL	Hollerith character used for plot
LABEL	Ten-character hollerith label
MODE = 1	Plots and prints the function Y(X)
MODE = 2	Is used for accumulating functions when plots of more than one function are desired
MODE = 3	Is used for the last accumulated function to print the entire set
MODE = 0	Plots and labels the grid only, and should be used prior to the first MODE = 2 call

9. Listing

```
PROGRAM ZMODE(INPUT,OUTPUT)

C
C COMPUTES THE NORMAL-MODE INTERNAL-WAVE OSCILLATIONS AND
C DISPERSION RELATIONS OF THE UPPER-OCEAN THERMOCLINE.
C
  REAL A(8),NQUN(8)
  DATA NQUN/6HTCLINE,6HZMODES,4HUTIL/
  3 PRINT 99
  99 FORMAT(1H1////////30X,4H*** RDA PROGRAM ZMODE, VERSION JULY 1973 ***
    *////////)
  1 READ 100,(A(I),I=1,8)
  100 FORMAT(8A10)
C
  DO 2 I=1,3
    IF(A(2),EQ,NQUN(I)) GO TO (10,20,30),I
  2 CONTINUE
    IF(A(1),EQ,2HGO) GO TO 3
    IF(A(1),EQ,4HSTOP) STOP
    PRINT 98,(A(I),I=1,8)
  98 FORMAT(32X,2H* ,A10)
    GO TO 1
C
  10 CALL TCLINE(A)
    GO TO 1
C
  20 CALL ZMODES(A)
    GO TO 1
C
  30 CALL UTIL(A)
    GO TO 1
  END
```

```

SUBROUTINE TCLINE(A)
COMMON/EIGEN/N,DZ,DZQ,ZA,ZR,QN(400),F(200),FZ(200),G(200),GZ(200),
  HN,ERR
  DIMENSION A(8),VERR(5),ZZ(400),TT(400),TA(200),TDATA(200),
  * ZDATA(200),CYN(400)
  DATA VERR,PI,GRAV,CT1,CT2/5HINPUT,3HSET,4HEXEC,5HPRINT,5HGRAPH,
  * 3.141592653,9.82,-.6E-4,.15F-4/

C
  DO 1 I=1,5
    IF(A(1),EQ,VERR(I)) GO TO (10,20,30,40,50),I
  1 CONTINUE
  PRINT 99, A(1)
99 FORMAT(20X,31HILLEGAL INSTRUCTION, TCLINE * ,A10,2H *)
  RETURN

C
C   INPUT TEMPERATURE PROFILE
C
  10 DECODE(10,100,A(4)) NDATA
  100 FORMAT(I10)
  READ 101,(ZDATA(I),TDATA(I),I=1,NDATA)
  101 FORMAT(2F10.0)
  RETURN

C
C   SET THERMOCLINE AND INTEGRATION PARAMETERS
C
  20 DECODE(30,200,A(4)) ZA,ZR,N
  200 FORMAT(2F10.0,I10)
  DZ=ZA/(N-1)
  DZQ=DZ**2
  N2=2*N-1
  RETURN

C
C   COMPUTE INTERPOLATED STABILITY PROFILE
C
  30 CALL SPLINES(NDATA,ZDATA,TDATA,TA)
  J=1
  DO 35 I=1,N2
    Z=DZ*.5*(I-1)
    Z=AMAX1(Z,ZDATA(1))
    CALL SPLINE2(Z,T,DT,ZDATA,TDATA,TA,NDATA,J)
    ZZ(I)=Z
    TT(I)=T
    QN(N2+1-I)=-GRAV*(CT1+CT2*T)*DT
  35 CONTINUE

C
  SUM=0.
  DO 36 I=1,N2
    QN1=SQRT(AMAX1(QN(I),0.))
    CYN(N2+1-I)=QN1*1800./PI
  36 SUM=SUM+QN1
  HN=.5*DZ*SUM
  RETURN

```

C  
C  
C

PRINT TEMPERATURE AND STABILITY PROFILES

```
40 DECODE(10,100,A(4)) ISKIP
   IF(ISKIP.EQ.0) ISKIP=4
   KOUNT=-1
   DO 45 I=1,N2,ISKIP
   KOUNT=KOUNT+1
   IF(MOD(KOUNT,50).EQ.0) PRINT 401
401 FORMAT(1H1//30X,27H*** THERMOCLINE PROFILE ***//
* 25X,8HDEPTH, M,AX,7HTEMP, C,4X,11HN(Z), CY/HR/)
   PRINT 402,ZZ(I),TT(I),CYN(I)
402 FORMAT(17X,3F15.2)
45 CONTINUE
   RETURN
```

C  
C  
C

GRAPH PROFILES

```
50 DECODE(30,500,A(4)) T1,T2,WSCALE
500 FORMAT(3F10.0)
   TSCALE=T1-T2
   MD=1
   SYM=1HT
   LBL=7HPROFILE
   CALL GRAPH(TT,ZZ,T2,-ZA,TSCALE,ZA,N2,MD,SYM,LBL)
   SYM=2HN
   CALL GRAPH(CYN,ZZ,0,-ZA,WSCALE,ZA,N2,MD,SYM,LBL)
   RETURN
   END
```

```

SUBROUTINE ZMODES(A)
  DIMENSION A(8),VERB(5),X(200),Y(200)
  COMMON/FLD/TK(51),TGAMMA(51,20),TDGAMMA(51,20),FT(51,20,2),
  . FIZ(51,20,2),Z1,Z2,NK
  COMMON/EIGEN/N,DZ,DZ0,ZA,ZB,ON(400),F(200),FZ(200),G(200),GZ(200),
  . HN,ERR
  REAL K,KMAX,NORM,KCY
  DATA VERB,FUN,PI/SHINPUT,3HSET,4HEXEC,5HPRINT,5HGRAPH,9HFUNCTIONS,
  . 3.141592653A/

C
  DO 1 I=1,5
    IF(A(I).EQ.VERB(I)) GO TO (10,20,30,40,50),I
  1 CONTINUE
  PRINT 100,A(1)
100 FORMAT(10X,30HILLEGAL INSTRUCTION, ZMODES **,A10,2H**)
  RETURN

C
C
  10 CONTINUE
  RETURN

C
C
  SET WAVENUMBER SCALE AND RESOLUTION

  20 DECODE(30,200,A(4)) KMAX,NK,ERR
200 FORMAT(F10.0,I10,E10.0)
  IF(ERR.EQ.0.) ERR=1,E=A
  KMAX=.002*PI*KMAX
  DK=KMAX/(NK-1)
  DO 21 IK=1,NK
21 TK(IK)=DK*(IK-1)
  RETURN

```



```

C
C      MAIN COMPUTATION OF MODE FUNCTIONS AND EIGENVALUES
C
30  DECODE (40,300,A(4)) M1,M2,Z1,Z2
300  FORMAT(2I10,2F10.0)
      NF1=N+.5=Z1/DZ
      NF2=N+.5=Z2/DZ
      PRINT 301,NK
301  FORMAT(1H1////28X,33H*** EIGENVL COMPUTATION TIMES ***,//
* 30X,12H* RESOLUTION,13,14H VALUFS/MODE *////
* 31X,26HMODE  AVG ITERATIONS  TIME,/)
C
      DO 35 MODE=M1,M2
      STIME=SECOND(X)
      GAMMA=(MODE-.5)*PI/HN
      DG=DDG=0.
      AVIT=0.
C
      DO 34 IK=1,NK
      K=TK(IK)
      GAMMA=GAMMA+DK*(DG+.5*DDG)
      GAMMA0=GAMMA
      DG1=DG
      CALL EIGENVL(GAMMA,K,NORM,ONORM,MODE,MERR,IERR,IT)
      IF(MERR.EQ.0.AND.IERR.EQ.0) GO TO 33
      KCY=500.*K/PI
      PRINT 302,MODE,KCY,MERR,IERR
302  FORMAT(/40X,29HEIGENVL FAILED TO CONVERGE AT,/
* 45X,5HMODE ,12,/45X,11HWAVENUMBER ,F 8.2/
* 45X,7HMERR = ,12/45X,7HIERR = ,12)
      STOP
C
33  R=NORM/ONORM
      DG=K*R/GAMMA
      DDG=DDG+DG
      IF(IK.EQ.1) DDG=DK*R/GAMMA
      TGAMMA(IK,MODE)=GAMMA
      TDGAMMA(IK,MODE)=DG
      RNORM=SQRT(ONORM)
      FI(IK,MODE,1)=F(NF1)/RNORM
      FI(IK,MODE,2)=F(NF2)/RNORM
      FIZ(IK,MODE,1)=FZ(NF1)/(DZ*RNORM)
      FIZ(IK,MODE,2)=FZ(NF2)/(DZ*RNORM)
      AVIT=AVIT+IT
34  CONTINUE
C
      TIME=SECOND(X)-STIME
      AVIT=AVIT/NK
      PRINT 303,MODE,AVIT,TIME
303  FORMAT(32X,12,F12.2,F11.4)
35  CONTINUE
      RETURN

```

```
C
C      PRINT DISPERSION RELATIONS
C
40  DECODE(20,300,A(4))M1,M2
    DO 45 MODE=M1,M2
      PRINT 400,MODE
400  FORMAT(1H1//30X,9H*** MODE ,12,25H DISPERSION RELATIONS ***//
* 20X,8HK, CY/KM,4X,8HW, 1/SEC ,4X,8HW, CY/HR,4X,8HC, M/SEC,3X,
* 9HCG, M/SFC,/)
C
    DO 44 IK=1,NK
      K=TK(IK)
      C=1./TGAMMA(IK,MODE)
      W=C*K
      CG=C*(1.-C*K*TGAMMA(IK,MODE))
      KCY=500.*K/PI
      WHR=3.6*C*KCY
      PRINT 401, KCY,W,WHR,C,CG
401  FORMAT(16X,5F12,5)
44  CONTINUE
45  CONTINUE
    RETURN
C
C      GRAPH DISPERSION CURVES
C
50  DECODE(10,500,A(3)) TYPE
500  FORMAT(A10)
    IF(TYPE,EQ,FUN) GO TO 60
    LBL=10HDISPERSTON
    DECODE(40,501,A(4)) M1,M2,WSCALE,SKALE
501  FORMAT(2I10,2F10,0)
    SCALE=.002*PI*SKALE
    MD=0
    CALL GRAPH(X,Y,0.,0.,SKALE,WSCALE,200,MD,SYM,LBL)
    MD=2
    DKK=KMAX/199.
```

C

```

DO 55 MODE=M1,M2
MODE1=MOD(MODE,10)
ENCODE(1,502,SYM) MODE1
502 FORMAT(I1)
IF(MODE.EQ.M2) MD=3
IK=1
DO 52 J=1,200
K=DKK+(J-1)
X(J)=K
GAMMA=FINT(K,TK,TGAMMA(1,MODE),TDGAMMA(1,MODE),NK,IK)
52 Y(J)=1800.*K/(GAMMA*PI)
CALL GRAPH(X,Y,0.,0.,SCALE,WSCALE,200,MD,SYM,LBL)
55 CONTINUE
RETURN

```

C  
C  
C

GRAPH NORMAL MODE FUNCTIONS

```

60 DECODE(50,600,A(4)) M1,M2,MULT,FSCALE,ZSCALE
600 FORMAT(3I10,2F10.0)
DO 65 MODE=M1,M2
SCALE=1,
FO=0,
ZO=ZSCALE
SYM=1H*
ENCODE(10,601,LBL) MODE
601 FORMAT(5HMODE,I2)
DELK=KMAX/(MULT-1)
DO 61 J=1,N
61 Y(J)=DZ*(J-1)-ZA
MD=0
CALL GRAPH(F,Y,FO,ZO,SCALE,ZSCALE,N,MD,SYM,LBL)
MD=2
IK=1

```

C

```

DO 65 I=1,MULT
K=DELK*(I-1)+.999
GAMMA=FINT(K,TK,TGAMMA(1,MODE),TDGAMMA(1,MODE),NK,IK)
CALL EIGENVL(GAMMA,K,NORM,ONORM,MODE,HERR,IERR,IT)
SCALE=FSCALE*SQRT(NORM/ZA)
FO=SCALE*(.25+.5*(I-1)/(MULT-1))
IF(I.EQ.MULT) MD=3
CALL GRAPH(F,Y,FO,ZO,SCALE,ZSCALE,N,MD,SYM,LBL)
65 CONTINUE
RETURN
END

```

```
SUBROUTINE EIGENVL(GAMMA,K,NORM,GNORM,MODE,MERR,IERR,IT)
COMMON/EIGEN/N,DZ,DZQ,ZA,ZR,QN(400),F(200),F7(200),G(200),GZ(200),
HN,ERR
REAL K, NORM
```

```
C
MERR=0
IERR=0
DGMAX=.6/HN
SGN=1.
IF(MOD(MODE,2).EQ.0) SGN=-1.
```

```
C
C
C ITERATE TO EIGENVALUE
```

```
DO 10 I=1,10
CALL EIGENFN(GAMMA,K,v,WG)
IF( SGN*WG.LE.0.) GO TO 11
DGAMMA=W/(2.*GAMMA*WG)
IF(ABS(DGAMMA).GT.DGMAX) DGAMMA=SIGN(DGMAX,DGAMMA)
GAMMA=GAMMA-DGAMMA
IF(ABS(DGAMMA/GAMMA).LE.FRR) GO TO 12
10 CONTINUE
```

```
C
11 IERR=1
C
C WRONSKIAN SLOPE HAS WRONG SIGN
```

```
C
C CHECK FOR CORRECT MODE NUMBER
```

```
C
12 M=1
IT=1
EPS=DZQ*W/WG
FZ(N)=FZ(N)-EPS*GZ(N)
DO 20 I=2,N
F(I)=F(I)-EPS*G(I)
FZ(I)=FZ(I)-EPS*GZ(I)
20 CONTINUE
N1=N-1
DO 25 I=2,N1
IF((F(I-1)*F(I).LT.0.).OR,F(I).EQ.0.) M=M+1
25 CONTINUE
MERR=M-MODE
IF(MERR.NE.0.OR,IERR.EQ.1) RETURN
```

C  
C  
C

COMPUTE NORMALIZATION FACTOR

```
QNMORM=WG*FZ(N)/DZ
SUM=0.
DO 30 I=2,N
30 SUM=SUM+F(I)**2
NORM=DZ*(SUM+.5*F(1)**2+F(1)*D7/6.)
IF(ZB,EQ,ZA) RETURN
IF(K,EQ,0.) GO TO 50
X=K*(ZB-ZA)
IF(X,GT,100.) GO TO 40
EX=EXP(X)
SINH0=(EX-1./EX)**2*.25
EX=EX**2
NORM=NORM+ (.125*(EX-1./EX)/K+.5*(ZB-ZA))*F(1)**2/SINH0
RETURN
40 NORM=NORM+.5*F(1)**2/K
C
RETURN
C
50 NORM=NORM+F(1)**2*(ZB-ZA)/3.
RETURN
END
```



```
FUNCTION FINT(Z,ZT,FT,FTZ,N,I)  
DIMENSION ZT(1),FT(1),FTZ(1)
```

C

```
3 IF(Z,LT,ZT(I)) GO TO 1  
4 IF(Z,GT,ZT(I+1)) GO TO 2
```

C

```
DZ=ZT(I+1)-ZT(I)  
U=(Z-ZT(I))/DZ  
V=1.-U  
FINT=U*FT(I+1)+V*FT(I)-U*V*((U+V)*(FT(I)-FT(I+1))+DZ*(U*FTZ(I+1)  
* -V*FTZ(I)))  
RETURN
```

C

```
2 I=I+1  
IF(I,LT,N) GO TO 4  
I=-1  
RETURN
```

C

```
1 I=I-1  
IF(I,GT,1) GO TO 3  
I=-1  
RETURN  
END
```

```
      SUBROUTINE SPLINES(N,Z,C,A)
C
C      SUBROUTINE SPLINES CALCULATES THE CUBIC SPLINE COEFFICIENTS,
C      CALLING ARGUMENTS
C      N  NUMBER OF POINTS, NOT TO EXCEED 100
C      Z  VECTOR OF INPUT DATA FOR INDEPENDENT VARIABLE
C      C  VECTOR OF INPUT DATA FOR DEPENDENT VARIABLE
C      A  COEFFICIENTS OF SPLINE
C
      DIMENSION Z(1),C(1),A(1),Y(100),DC(100),DZ(100)
      A(1)=0.
      A(N)=0.
      N1=N+1
C
C      SET UP DIFFERENCES,
C
      DO 12 I=1,N1
      DZ(I)=Z(I+1)-Z(I)
      DC(I)=(C(I+1)-C(I))/DZ(I)
12  CONTINUE
      Y(1)=0.
      IF(N1.LT.2) GO TO 17
C
C      SOLVE TRIDIAGONAL SYSTEM,
C
      DO 15 I=2,N1
      PIV=2.0*(DZ(I-1)+DZ(I))-DZ(I-1)*Y(I-1)
      Y(I)=DZ(I)/PIV
15  A(I)=(DC(I)-DC(I-1)-DZ(I-1)*A(I-1))/PIV
      DO 16 IB=2,N1
      I=N+1-IB
16  A(I)=A(I)-Y(I)*A(I+1)
17  RETURN
```



```
SUBROUTINE SPLINE(Z,F,FZ,ZT,FT,A,N,I)  
DIMENSION ZT(1),FT(1),A(1)
```

C

```
M=0
```

```
3 IF(Z.LT,ZT(I)) GO TO 1
```

```
4 IF(Z.GT,ZT(I+1)) GO TO 2
```

C

```
DZ=ZT(I+1)-ZT(I)
```

```
U=(Z-ZT(I))/DZ
```

```
V=1.-U
```

```
F=U*FT(I+1)+V*FT(I)-DZ**2*U*V*(A(I)*(1.+V)+A(I+1)*(1.+U))
```

```
IF(M.EQ.0) RETURN
```

```
FZ=(FT(I+1)-FT(I))/DZ+DZ*(A(I)*(1.-3.+V**2)-A(I+1)*(1.-3.+U**2))
```

```
RETURN
```

C

```
2 I=I+1
```

```
IF(I.LT,N) GO TO 4
```

```
I=1
```

```
RETURN
```

C

```
1 I=I-1
```

```
IF(I.GE.1) GO TO 3
```

```
I=1
```

```
RETURN
```

C

```
ENTRY SPLINE2
```

```
M=1
```

```
GO TO 3
```

```
END
```

```
SUBROUTINE GRAPH(X,Y,X0,Y0,XSCALE,YSCALE,N,MODE,SYMBOL,LABEL)
LOGICAL XOUT,YOUT
DIMENSION X(1),Y(1),A(11,51),XLABL(6),MASK(10)
DATA GRID,FRAME,ENDS,BLANK/10H          +,10H-----+,
*10H          ,10H          /
IF(MODE.GT.1) GO TO 2
```

C  
C  
C

INITIALIZE AND PLOT GRID

```
1 DO 3 I=1,51
  DO 3 K=1,11
3  A(K,I)=BLANK
  DO 4 K=2,11
    A(K,1)=FRAME
4  A(K,51)=FRAME
  DO 5 I=2,50
    A(1,I)=ENDS
5  A(11,I)=ENDS
  DO 6 I=11,41,10
    DO 6 K=2,11
6  A(K,I)=GRID
  DO 7 K=1,6
    XLABL(K)=X0+0.2*XSCALE*(K-1)
    YLABL    =Y0+0.2*YSCALE*(K-1)
    LINE=61-10*K
50 FORMAT(F9.3,1H+)
7  ENCODE(10,50,A(1,LINE))YLABL
  MASK(10)=63
  DO 8 K=1,9
8  MASK(10-K)=N.77R,A.SHIFT(MASK(11-K),6)
  IF(MODE.EQ.0)RETURN
```

C  
C  
C

SYMBOL PLOT

```
2 SYMBOL=SHIFT(SYMBOL,-54).A.77B
  SYM=SYMBOL
  DO 9 K=1,9
9  SYM=SYMBOL.O.(SHIFT(SYM,6).A..N.77B)
  SYMBOL=SYM
  DO 10 I=1,N
  LINE=50.*(1.-(Y(I)-Y0)/YSCALE)+1.5
  KOLUM=100.*(X(I)-X0)/XSCALE+10.5
  YOUT=LINE.LT.1.OR.LINE.GT.51
  XOUT=KOLUM.LT.10.OR.KOLUM.GT.110
  IF(XOUT.OR.YOUT) GO TO 10
  KWORD=(KOLUM-1)/10
  KHAR=KOLUM-10*KWORD
  KWORD=KWORD+1
  WORD=A(KWORD,LINE).A..N.MASK(KHAR)
  A(KWORD,LINE)=WORD.O.(SYM,A.MASK(KHAR))
10 CONTINUE
  IF(MODE.EQ.2) RETURN
```

C  
C  
C

PRINT ACCUMULATED GRAPHS

```
PRINT 1000,LABEL,((A(K,L),K=1,11),L=1,51)
1000 FORMAT(1H1///15X,A10/(5X,11A10))
PRINT 2000,(X,ABL(K),K=1,6)
2000 FORMAT(/6X,6(F13.4,7X))
RETURN
END
```

```
SUBROUTINE UTIL(A)  
COMMON/FLD/TK(51),TGAMMA(51,20),TOGAMMA(51,20),FI(51,20,2),  
FIZ(51,20,2),Z1,Z2,NK  
DIMENSION A(A)  
RETURN  
END
```